A

Synopsis of

Proposed research work on

"Design and Development of Algorithms for Big data Classification"

by

Mr. Mahendra P. Deore

Under the guidance of

Dr. U. V. Kulkarni

Professor, Computer Science & Engineering

S.G.G.S. Institute of Engineering and Technology,

Vishnupuri, Nanded – 431 606, India,

E-mail: uvkulkarni@sggs.ac.in, Fax: 02462-229236

1. Research Title:

Design and Development of Algorithms for Big data Classification

2. Research Guide:

Dr. U. V. Kulkarni

Professor of Computer Science & Engineering

3. Area of Research:

Big Data

4. Abstract:

Now a day's classification of huge amount growing data sets has become challenging. This research focuses the challenges introduced by the Big Data classification associated with the malware detectors and classifiers. The number of new malware specimen has an exponential growth which cannot be analyzed fast enough. The Classification of malware samples based on their behaviors is necessary for the computer security. This necessity has motivated the development of a new model that is efficient in categorizing malware variations into their definite family groups. This model is presented and discussed in this research, where importance is given to the extraction and selection of new features for the efficient representation of malware samples. The grouping of features can be done on the basis of different characteristics of malware behavior, and those could be combined together to performed their classification in to different malware families. The proposed research work is mainly focused to design an algorithm to achieve better accuracy level for Malware detection and classification in usual and unusual environments such as against evasion attacks or poisoning attacks which are more common against malware detectors and malware classifiers.

5. Technical Keywords:

Big Data, Classification, Computer Security, Machine learning, Malware family, Windows Malware, Microsoft Malware Classification Challenge

6. Introduction:

- **Big Data:**

  Big Data is defined in terms of volume, variety and velocity. In the Big Data Research the term Big Data Analytics is defined as the process of analyzing and understanding the characteristics of massive size data sets by extracting useful statistical patterns.

- **Malware:**

  Malware is an abbreviation of "malicious software". This software used to disrupt computer operations, gain access to private systems, collect sensitive information. The first category of malware concerns parasitic software fragments that attach themselves to some existing executable content. The fragment may be machine code that infects some existing application, utility, or system program, or even the code

used to boot a computer system, Malware is defined by its malicious intent, acting against the requirements of the user.

- Recently malware is developed in complicated techniques to avoid traditional as well as modern malware protection mechanisms. On the other side, anti-malware solutions need to develop counter mechanisms for detecting and neutralizing them.
- The huge number of malware families, and malware variants inside the families, causes a major problem for antimalware products,[1]. For example, McAfee Lab's antimalware solutions reported more than 350M total unique malware samples in Q4 of 2014, that represents a growth of 17% with respect to the analogous data in Q3,[2].
- Symantec reported more than 44.5 million new pieces of malware created in May 2015,[3].
- Analyzing the malicious intent in this vast amount of data requires a huge effort by anti malware companies.
- One of the main reasons for this high volume of malware samples is the extensive use of polymorphic and metamorphic techniques by malware developers, which means that malicious executable files belonging to the same malware family are constantly modified and/or unintelligible.
- In particular, polymorphic malware has a static mutation engine that encrypts and decrypts the code, while metamorphic malware automatically modify the code each time it is circulated.
- Malware detection and classification techniques are two separate tasks that are performed by anti-malware companies.
- Firstly, an executable needs to be analyzed to detect if it reveals any malicious content: Then, in the case a malware is detected, it is assigned to the most appropriate malware family through a classification mechanism.
- The analysis of malicious programs is usually carried out by static techniques [4, 5, 6] and dynamic techniques [7, 8, 9, 10].
- Analyzers extract various characteristics from the programs' syntax and semantic such as operation codes [11] and function call graph [12] from the disassembled code, or string signatures [13] and byte code n-grams [14,15] from the hex code, or different structural characteristics from the PE header, such as dependencies between APIs [5] and DLLs [6].
- Some other works [16] also explored the analysis of metadata such as the number of bitmaps, the size of import and export address tables besides the PE header's content.
- The abovementioned content-based detection systems, like those considering byte code n-grams, APIs, and assembly instructions, are inherently susceptible to false detection due to the fact of polymorphism and metamorphism.
- The proposed research focuses a learning-based system which will uses different malware characteristics to effectively assign malware samples to their corresponding families without doing any to convert a same difficult program to understandable and unpacking process.

- Although unpacking may lead to the extraction of more valuable features if the packers are known, unpacking is a costly and dealing with customized packers is even more challenging.
- Hence aim to perform classification without the need to unpack the sample.
- For each malware sample, a set of content-based features will be computed not only by relying on modern mechanisms, but also extraction of powerful complementary statistical features that reflects the structure of portable executable (PE) files.
- The decision of not using more complex models like n-grams, sequences, bags or graphs can allow devising a simple, however effective, and efficient malware classification system.

7. Problem Statement:

The proposed research work is mainly focused to design an algorithm to achieve better accuracy level for Malware detection and classification in usual and unusual environments such as against evasion attacks or poisoning attacks which are more common against malware detectors and malware classifiers.

8. Motivation:

The number of new malware specimen has an exponential growth which cannot be analyzed fast enough. The Classification of malware samples based on their behaviors is necessary for the computer security which can gain access or damage a computer without the knowledge of the owner.

9. Objective:

- To design an algorithm to achieve better accuracy level for Malware detection and classification in usual and unusual environments.

10. Brief Literature Survey:

- In the research process literature review plays a very important role. It is a source from where research idea is drawn and to be developed into concepts and finally theories. Here in this literature survey, all primary, secondary and tertiary sources of information were searched.
- Prior to the development of signatures for anti-malware products, the two main tasks that have to be carried out within the scope of malware analysis are malware detection, and malware classification.
- While the goal of malware detection mechanisms is to catch the malware in the wild, malware classification systems assign each sample to the correct malware family.
- These systems can be roughly divided into two groups, based, respectively, on dynamic or static analysis.
- **Dynamic analysis:**
  - Researchers have put a lot of effort in proposing behavior-based malware detection methods that capture the behavior of the program at runtime. One way to observe the behavior of a program is to monitor the interactions of the program with the operating system through the analysis of the API calls [7, 8].

- In order to devise an effective and more robust system, some approaches considered additional semantic information like the sequence of the API calls [9], and the use of graph representations [18, 19, 20].
- These approaches monitor the program's behaviour by analyzing the temporal order of the API calls, and the effect of API calls on registers [21], or by extracting a behavioral graph based on the dependency between API call parameters. Additionally, in contrast to the above program-centric detection approaches, some proposals address the issue by a global, system-wide approach.
- For example, Lanzi et al. [22] proposed an access activity model that captures the generalized interactions of gentle applications with operating system resources, such as files and the registry, and then detects the malware with very a very low false positive rate.
- A recent survey on 36 research papers on dynamic analysis techniques [23] pointed out that the common shortcomings of dynamic analysis techniques are the problematic and somewhat obscure assumptions regarding the use of execution-driven datasets, and the lack of details and motivation on the security precautions that have been taken during the experimental phase.
- Moreover, recent malware is shipped with dynamic anti-analysis defenses that hide the malicious behaviour in the case a dynamic analysis environment is detected [24] and the lack of code coverage, as dynamic analysis is not designed to explore all or, at least, multiple execution paths of an executable [25].

- **Static analysis**
- Static approaches perform the analysis without actually executing the program.
- The research literature exhibits a large variety of static analysis methods.
- SAFE [26] and SAVE [27] have been among the most inuential approaches in heuristic static malware detection, as these works inspired many researchers in this area.
- The above two works proposed the use of different patterns to detect the presence of malicious content in executable files. Since that time, a large variety of techniques have been explored based on different malware attributes, such as the header of the PE, the body of the PE, or both of them.
- Analysis is further carried out either directly on the byte code [28, 15], or by disassembling the code and extracting opcodes and other relevant detailed information on the content of the program [11].
- The main issue in static analysis is coping with packing and obfuscation. Recently, some paper addressed this issue by proposing a generic approach for the automatic deobfuscation of obfuscated programs without making any assumption about the obfuscation technique [29].
- Static techniques have been also employed to assess if a malware detected in the wild is similar to a previously seen variant, without actually performing the costly task of unpacking [30, 31].

- All of the malware detection and malware classification systems rely on the extraction of either static or dynamic features. So, basically, the same features used for malware detection are used for malware classification purposes.
- As this research focuses on malware classification based on the extraction of static features.

11. Proposed Research Work:

- This research focuses on malware classification; the most relevant issue is related to the choice of the features that will be used to represent each malware sample for the classification task.
- To attain accurate and fast classification results different types of features could be integrated such as content-based features as well as structural features.
- Figure 1 shows block diagram of overall architecture to be followed in throughout research.

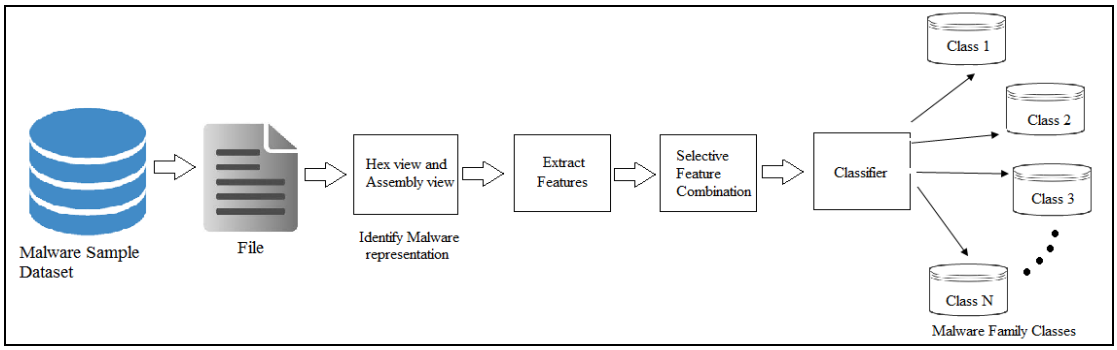

Figure 1. Proposed Architecture

- Following is the planed methodology to attain the goal of research:

- **Malware representation**

- Two common representations of a malware sample are by the hex view, and the assembly view.
- The hex view represents the machine code as a sequence of hexadecimal digits, which is the accumulation of consecutive 16-bytes words.
- The task of disassembling a binary executable into its sequence of assembly instructions can be performed by two main techniques, namely by the linear sweep algorithm, and the recursive traversal algorithm [17].

- **Features**

- For accurate and fast classification, features can be extracted from both hex view, and from the assembly view to exploit complementary information brought by these two representations.
- This complementary information is usually related to the essence of maliciousness and the experimental results will show how the combination of information from the two views can help improving the effectiveness of the whole system.

- **Feature combination**

  The selective features will be combined together and will be designed as a combined feature categories to load all the feature categories in a single then same will be used to run a classifier on them.

- **Classification**

  In earlier period numbers of classification techniques have been proposed and the selection of the most appropriate classifier for a given task is often guided by previous experience in different domains, as well as by trial & error procedures. However, recently some researchers have concluded that random forests and SVM are the two classification mechanisms to produce good performances [19] also some researchers have used the XGBoost technique [8], which is a parallel implementation of the gradient boosting tree classifier. In addition, bagging [15] can also be considered to boost this single combined feature model, which is effortless, classifier independent, and an efficient method to improve the classification quality.

12. References:

[1]. Mansour Ahmadi, Dmitry Ulyanov, Stanislav Semenov, Mikhail Trofimov, Giorgio Giacinto, "Novel Feature Extraction, Selection and Fusion for Effective Malware Family Classification", Conference Paper: 6th ACM Conference on Data and Applications Security and Privacy (CODASPY), At New Orleans, USA, March 2016

[2]. Mcafee labs threats report, february. http://www.mcafee.com/us/resources/reports/rp-quarterly-threat-q4-2014.pdf, 2015.

[3]. Symantec intelligent report, may. https://www.symantec.com/content/en/us/enterprise/other_resources/intelligence_report_05-2015.en-us.pdf, 2015.

[4]. A. Sami, B. Yadegari, H. Rahimi, N. Peiravian, S. Hashemi, and A. Hamze. Malware detection based on mining api calls. In Proceedings of the 2010 ACM Symposium on Applied Computing, SAC '10, pages 1020-1025, New York, NY, USA, 2010. ACM.

[5]. Y. Ye, D. Wang, T. Li, D. Ye, and Q. Jiang. An intelligent pe-malware detection system based on association mining. Journal in Computer Virology, 4(4):323-334, 2008.

[6]. M. Narouei, MansourAhmadi, G. Giacinto, H. Takabi, and A. Sami. Dllminer: Structural mining for malware detection. Security and Communication Networks, 2015.

[7]. C. Willems, T. Holz, and F. Freiling. Toward automated dynamic malware analysis using cwsandbox. Security Privacy, IEEE, 5(2):32-39, March 2007.

[8]. K. Rieck, T. Holz, C. Willems, P. Dussel, and P. Laskov. Learning and classi_cation of malware behavior. In Proceedings of the 5th International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment, DIMVA '08, pages 108-125, Berlin, Heidelberg, 2008. Springer-Verlag.

[9]. M. Ahmadi, A. Sami, H. Rahimi, and B. Yadegari. Malware detection by behavioural sequential patterns. Computer Fraud & Security, 2013(8):11-19, 2013.

[10]. T. Wuchner, M. Ochoa, and A. Pretschner. Malware detection with quantitative data ow graphs. In Proceedings of the 9th ACM Symposium on Information, Computer and Communications Security, ASIA CCS '14, pages 271-282, New York, NY, USA, 2014. ACM.

[11]. I. Santos, F. Brezo, X. Ugarte-Pedrero, and P. G. Bringas. Opcode sequences as representation of executables for data-mining-based unknown malware detection. Information Sciences, 231(0):64-82, 2013. Data Mining for Information Security.

[12]. X. Hu, T.-c. Chiueh, and K. G. Shin. Large-scale malware indexing using function-call graphs. In

Proceedings of the 16th ACM Conference on Computer and Communications Security, CCS '09, pages 611-620, New York, NY, USA, 2009. ACM.

[13]. K. Gri_n, S. Schneider, X. Hu, and T.-C. Chiueh. Automatic generation of string signatures for malware detection. In Proceedings of the 12th International Symposium on Recent Advances in Intrusion Detection, RAID '09, pages 101-120, Berlin, Heidelberg, 2009. Springer-Verlag.

[14]. S. M. Tabish, M. Z. Sha_q, and M. Farooq. Malware detection using statistical analysis of byte-level _le content. In Proceedings of the ACM SIGKDD Workshop on Cyber Security and Intelligence Informatics, CSI-KDD '09, pages 23-31, New York, NY, USA, 2009. ACM.

[15]. Novel active learning methods for enhanced {PC} malware detection in windows {OS}. Expert Systems with Applications, 41(13):5843-5857, 2014.

[16]. M. Sha_q, S. Tabish, F. Mirza, and M. Farooq. Pe-miner: Mining structural information to detect malicious executables in realtime. In E. Kirda, S. Jha, and D. Balzarotti, editors, Recent Advances in Intrusion Detection, volume 5758 of Lecture Notes in Computer Science, pages 121-141. Springer Berlin Heidelberg, 2009.

[17]. B. Schwarz, S. Debray, and G. Andrews. Disassembly of executable code revisited. In Proceedings of the Ninth Working Conference on Reverse Engineering (WCRE'02), WCRE '02, pages 45-, Washington, DC, USA, 2002. IEEE Computer Society.

[18]. C. Kolbitsch, P. M. Comparetti, C. Kruegel, E. Kirda, X. Zhou, and X. Wang. Effective and efficient malware detection at the end host. In Proceedings of the 18th Conference on USENIX Security Symposium,SSYM'09, pages 351-366, Berkeley, CA, USA, 2009. USENIX Association.

[19]. M. Fredrikson, S. Jha, M. Christodorescu, R. Sailer, and X. Yan. Synthesizing near-optimal malware specifications from suspicious behaviors. In Proceedings of the 2010 IEEE Symposium on Security and Privacy, SP '10, pages 45-60, Washington, DC, USA, 2010. IEEE Computer Society.

[20]. F. Karbalaie, A. Sami, and M. Ahmadi. Semantic malware detection by deploying graph mining. International Journal of Computer Science Issues, 9(1), 2012.

[21]. M. Ghiasi, A. Sami, and Z. Salehi. Dynamic vsa: a framework for malware detection based on register contents. Engineering Applications of Artificial Intelligence, 44:111-122, 2015.

[22]. A. Lanzi, D. Balzarotti, C. Kruegel, M. Christodorescu, and E. Kirda. Accessminer: Using System-centric models for malware protection. In Proceedings of the 17th ACM Conference on Computer and Communications Security, CCS '10, pages 399-412, New York, NY, USA, 2010. ACM.

[23]. C. Rossow, C. Dietrich, C. Grier, C. Kreibich, V. Paxson, N. Pohlmann, H. Bos, and M. van Steen. Prudent practices for designing malware experiments: Status quo and outlook. In Security and Privacy (SP), 2012 IEEE Symposium on, pages 65-79, May 2012.

[24]. J. Qiu, B. Yadegari, B. Johannesmeyer, S. Debray, and X. Su. A framework for understanding dynamic anti-analysis defenses. In Proceedings of the 4th Program Protection and Reverse Engineering Workshop, PPREW-4, pages 2:1-2:9, New York, NY, USA, 2014. ACM.

[25]. A. Moser, C. Kruegel, and E. Kirda. Exploring multiple execution paths for malware analysis. In Proceedings of the 2007 IEEE Symposium on Security and Privacy, SP '07, pages 231-245, Washington, DC, USA, 2007. IEEE Computer Society.

[26]. M. Christodorescu and S. Jha. Static analysis of executables to detect malicious patterns. In Proceedings of the 12th Conference on USENIX Security Symposium - Volume 12, SSYM'03, pages 12-12, Berkeley, CA, USA, 2003. USENIX Association.

[27]. A. H. Sung, J. Xu, P. Chavez, and S. Mukkamala. Static analyzer of vicious executables (save). In

Proceedings of the 20th Annual Computer Security Applications Conference, ACSAC '04, pages 326-334, Washington, DC, USA, 2004. IEEE Computer Society.

[28]. S. M. Tabish, M. Z. Sha q, and M. Farooq. Malware detection using statistical analysis of byte-level le content. In Proceedings of the ACM SIGKDD Workshop on Cyber Security and Intelligence Informatics, CSI-KDD '09, pages 23-31, New York, NY, USA, 2009. ACM.

[29]. B. Yadegari, B. Johannesmeyer, B. Whitely, and S. Debray. A generic approach to automatic deobfuscation of executable code. In IEEE Security and Privacy. IEEE, 2015.

[30]. G. Jacob, P. M. Comparetti, M. Neugschwandtner, C. Kruegel, and G. Vigna. A static, packer-agnostic filter to detect similar malware samples. In Proceedings of the 9th International Conference on Detection of Intrusions and Malware, and Vulnerability Assessment , DIMVA'12, pages 102-122, Berlin, Heidelberg, 2013. Springer-Verlag.

[31]. M. Narouei, Mansour Ahmadi, G. Giacinto, H. Takabi, and A. Sami. Dllminer: Structural mining for malware detection. Security and Communication Networks, 2015.

Mr. Mahendra P Deore

Research Scholar

Dr. U.V. Kulkarni

Guide
Professor Dept of CSE
SGGSIE&T, Nanded